Instructions for deploying rotten-potatoes to Heroku

This document provides instructions for deploying the rotten-potatoes app to Heroku in preparation for HW4. Heroku is a cloud-based Platform-as-a-Service (PaaS) provider. That is to say, Heroku provides cloud-based deployment and hosting services for SaaS applications. Heroku and other similar PaaS services such as Microsoft Azure, Amazon Web Services, and CloudFoundry, make it very easy to deploy and maintain SaaS applications in the cloud.

Step 1: Sign-up for a Heroku account:
Go to `heroku.com` and sign up for a free account. Be sure to remember your password as you will need it in a little while. While you are on the Heroku site, spend a few minutes browsing around to learn about some of the features of Heroku.

Step 2: **BUILD BASE ROTTON POTATOES APP**: Copy the starter files from hw4-{githubid} from a clone of the GitHub classroom link provided, to your local environment. This repo contains resources that will allow you to build the base version of rotten-potatoes quickly.

Follow the instructions in '`CreateRottonPotatoesCommands.md`' To get the base rotten potatoes application configured.

with a few minor configuration tweaks to make it ready for Heroku deployment. Navigate to the top-level of your hw4-{githubid} directory and type the command:

Now run the Rails server on your local development envirionment by entering the command:

```
export PORT={pick some unique port number i.e. between 25000 and 62000}
export IP=localhost
rails server -p $PORT -b $IP
```

Note that the browser MAY display an ActiveRecord::PendingMigration Error. This is because the database migrations specified in the db/migrate directory have not been applied to the sqlite database. Initialize and seed the development (sqlite) database by typing the following commands:

```
rake db:migrate
rake db:seed
```

The first of these commands applies the schema specified in `db/migrate` to the development (sqlite) database and the second seeds the database with some initial values from the file `db/seeds.rb`

Now test your app in your local environment to make sure that everything is working properly in development mode. Once you have verified correct operation, you can kill the rails server and close the preview window.

Step 3: Now you are ready to prepare for deployment on Heroku.

We need to make a few modifications to our Gemfile to prepare for Heroku production deployment.

1) Make a production group with a production database by adding the following section near the bottom of your GemFile

```
group :production do
  # for Heroku production deployment
  # NOTE RAILS PRIOR TO 5.15 do not support the pg 1.0.0 module!
  # NOTE: You may need to research proper gem version compatibility for your environment.
  gem 'pg', '~> 0.21'
  gem 'rails_12factor'
end
```

2) Move the "`gem 'sqlite3'`" line to the "`group :development, :test`" section in the Gemfile
3) Add "ruby '2.6.6' to the top of your Gemfile. To lock in a specific version of ruby to be used on Heroku. '`bundle install`', `git add Gemfile.lock, git commit;`

IMPORTANT: before using Heroku, you will need to install/update the Heroku Command Line Interface in your environment. Follow the instructions available here: https://devcenter.heroku.com/articles/getting-started-with-ruby#set-up

Next, you will need to set up your ssh keys for Heroku via the command:

```
heroku keys:add ~/.ssh/id_rsa.pub  (path and file name may vary)
```

You will be prompted to enter your Heroku login credentials (email and password) and asked to confirm that you want to add the ssh key to Heroku.

Now, making sure that you are in the hw4-{githubid} directory, type the following command to set up your app for deployment to set up your app for deployment to Heroku:

```
heroku create hw4-{githubid}_fall2020 --buildpack heroku/ruby
```

The command will display a URL with a (perhaps whimsical) name, something like:

```
https://murmuring-shore-69437.herokuapp.com
```

This is the URL for your app once it is deployed on Heroku, so be sure to copy it or write it down.

Step 4: Now, you are ready to deploy the app to Heroku. Heroku works with git, and pushing your app to Heroku is just like pushing it to GitHub. To push to Heroku, type the command:

```
git remote -v # See the currently defined git remote aliases
git push heroku master
```

When this command completes, Heroku will have attempted to deploy your app running in production mode on Heroku. As you work on HW4, you can push updates to Heroku at any time, just by doing another git push.**).**

IMPORTANT NOTE: The "git push heroku master" command, pushes the committed master branch of your local git repo to Heroku. If you edit, add, or delete files, you must commit them before pushing to Heroku. You can do this by typing:

```
git add .
git commit -m "message of your choosing"
```

Try accessing the app via a browser, using the Heroku URL that was provided by the "heroku create" command in step 3. Note that you will see an error message: "Something went wrong…". To find the source of the problem, lets do some detective work. Enter the command:

```
heroku logs
```

This command displays a log of actions related to your running app and is an excellent source of debugging information when something goes wrong. Note that the log shows an error message, complaining about an undefined movies table. This is because you have not yet set up the database schema for the production (Postgres) database on Heroku.

To set-up and seed the Heroku database, type:

```
heroku run rake db:schema:load
heroku run rake db:seed
```

Now, if everything has gone according to plan, you should see your running rotten-potatoes app, after refreshing the browser.

Congratulations. You have successfully deployed your first app to Heroku.